UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/681,064	12/15/2000	Andrew L. Bliss	MSFT-0218	9482

27372 7590 09/10/2004

WOODCOCK WASHBURN KURTZ
MACKIEWICZ & NORRIS LLP
ATTENTION: STEVEN J. ROCCI, ESQ.
ONE LIBERTY PLACE, 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER	
YIGDALL, MICHAEL J	
ART UNIT	**PAPER NUMBER**
2122	

DATE MAILED: 09/10/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/681,064	BLISS ET AL.	
	Examiner	Art Unit	
	Michael J. Yigdall	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 June 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-11,14-40 and 42-50* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-11,14-40 and 42-50* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This Office action is in reply to Applicant's response and amendment filed June 3, 2004. Claims 1-11, 14-40 and 42-50 remain pending.

Response to Arguments

2. Applicant's arguments have been fully considered but they are not persuasive.

3. Applicant contends that You does not teach elements recited in claim 1, and in particular does not teach a debugging type attribute selected from a plurality of debugging type attributes, a processor attribute selected from a plurality of processor attributes, a plurality of debugging type blocks, each debugging type block for supporting at least one of the debugging type attributes, and a plurality of processor blocks, each processor block for supporting at least one of the plurality of processor attributes (page 13, line 23 to page 14, line 8).

However, You does disclose determining the operating system of the debuggee (see column 72, lines 43-54), and similarly determining the architecture of the target processor from processor attributes provided by the debuggee (see column 72, line 55 to column 73, line 5). In other words, You teaches that the debuggee has a processor attribute selected from a plurality of processor attributes.

You further discloses a plurality of debugging types that are supported by the debugger (see column 6, lines 31-55). As presented in the previous Office action, it would have been apparent, i.e. obvious, to one of ordinary skill in the art at the time the invention was made to include a debugging type attribute, in addition to the processor attributes taught by You, to designate or select a debugging type from the plurality of supported debugging types. Such

attributes would, for example, enable the debugger to perform a type of debugging on a debuggee without the need for user input.

Moreover, You discloses that the debugger implements class hierarchies, in which classes are derived from the base classes to provide functionality for particular debuggees (see column 27, lines 13-24). The base classes are reusable and are not limited to any one processor architecture, operating system or runtime model (see column 66, lines 62-67). Importantly, derived classes are created according to the attributes of the debuggee (see column 73, lines 9-18). You further discloses a plurality of supported debugging types (see column 6, lines 31-55) and processor attributes (see column 9, lines 17-26) for which such classes would be derived. The derived classes are thus analogous to the "blocks" recited in the claims. Therefore, You teaches a plurality of debugging type blocks and a plurality of processor blocks for supporting the respective attributes.

4. In response to Applicant's argument that You does not fulfill a need for a single debugger engine that supports multiple debugging types and multiple processors, "such that supporting, updating and maintaining the single debugger engine is greatly simplified" (page 13, lines 4-9), the fact that Applicant has recognized another advantage which would flow naturally from following the suggestion of the prior art cannot be the basis for patentability when the differences would otherwise be obvious. See *Ex parte Obiaya*, 227 USPQ 58, 60 (Bd. Pat. App. & Inter. 1985). It should also be noted that although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-11, 14, 15, 18-29, 32, 34-38, 42 and 44-48 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,158,045 to You (herein "You").

With respect to claim 1 (original), You discloses a debugger for debugging any of a plurality of debuggees (see the title and abstract), each debuggee having a processor attribute selected from a plurality of processor attributes and representative of a type of processor associated with the debuggee (see column 72, line 55 to column 73, line 5, which shows the debuggee to have processor attributes from which the architecture or the type of the processor may be determined), the debugger being instantiated on a computer (see FIG. 1) and comprising:

(a) an engine for performing debugging functions with respect to any of the plurality of debuggees (see column 4, lines 41-50, which shows a portable debugging system having a server debugger object or engine that is responsible for any of a plurality of debugging environments), the engine including:

(i) a plurality of debugging type blocks (see column 27, lines 13-24 and column 66, lines 62-67, which shows that the debugger implements class hierarchies, in which classes are derived for particular debuggees, and column 73, lines 9-18, which shows

creating such derived classes, i.e. blocks), each debugging type block for supporting at least one of the plurality of debugging type attributes (see column 6, lines 31-55, which shows a plurality of debugging types supported by the engine, which would be implemented as derived classes or debugging type blocks); and

(ii) a plurality of processor blocks (see column 27, lines 13-24 and column 66, lines 62-67, which shows that the debugger implements class hierarchies, in which classes are derived for particular debuggees, and column 73, lines 9-18, which shows creating such derived classes, i.e. blocks), each processor block for supporting at least one of the plurality of processor attributes (see column 9, lines 17-26, which shows a plurality of processor attributes accounted for by the engine, which would be implemented as derived classes or processor blocks);

wherein a particular debugging type block and a particular processor block are selected for debugging a particular debuggee based on the debugging type attribute and processor attribute of the particular debuggee (see column 9, lines 28-33, which shows implementing or selecting an engine for a particular host environment, i.e. for a debuggee having a particular debugging type and particular processor attributes).

Although You discloses support for a plurality of debugging types (see column 6, lines 31-55) and a plurality of processor families or types (see column 9, lines 17-26), as well as attributes of the debuggee used to represent the processor type (see column 72, line 55 to column 73, line 5), You does not expressly disclose the limitation wherein each debuggee has a debugging type attribute selected from a plurality of debugging type attributes and representative of a type of debugging to be performed with respect to the debuggee.

However, it would have been apparent to one of ordinary skill in the art at the time the invention was made for each debuggee to include a debugging type attribute, in addition to the processor attributes, to designate a particular debugging type. Such an addition would, for example, enable the engine to perform the designated type of debugging without demanding a manual selection by the user.

With respect to claim 2 (original), You further discloses the limitation wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type (see column 10, lines 4-7, which shows that the portable debugging system uses abstraction, and column 9, lines 28-33, which shows that particular implementations may vary).

With respect to claim 3 (original), You further discloses the limitation wherein the debugging services include services selected from a group consisting of accessing memory, accessing context, accessing system information, inserting a breakpoint, removing a breakpoint, controlling execution, and combinations thereof (see column 10, lines 20-27, which shows the data exchanged between the debugging client and server for services such as stack or memory access, runtime or context information, breakpoints, and so on).

With respect to claim 4 (original), You further discloses the limitation wherein the debugging type abstraction comprises programming code, and wherein at least a portion of the programming code for the debugging type abstraction is common as between at least some debugging type blocks and is shared by such debugging type blocks (see column 10, lines 29-40,

which shows using C++ programming code and template classes to reuse or share common interfaces).

With respect to claim 5 (original), You further discloses the limitation wherein the programming code for the debugging type abstraction is organized into a tree form with generic code at a base node and more specific levels of code branching out at nodes therefrom, each debugging type block including at least one node from the tree (see FIG. 9, which shows the abstraction tree for address-related code; note that a similar tree would apply to the debugging type).

With respect to claim 6 (original), You further discloses the limitation wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor (see column 10, lines 4-7, which shows that the portable debugging system uses abstraction, and column 9, lines 28-33, which shows that particular implementations may vary).

With respect to claim 7 (original), You further discloses the limitation wherein the processor services include services selected from a group consisting of recognizing particular processor instructions, recognizing processor states, maintaining hardware breakpoints, assembling code for the processor, disassembling code from the processor, disassembling code from a dump file produced by the processor, and combinations thereof (see column 10, lines 20-27, which shows the data exchanged between the debugging client and server for services such as register or processor state information, breakpoints, hardware exceptions, and so on).

With respect to claim 8 (original), You further discloses the limitation wherein the processor abstraction comprises programming code, and wherein at least a portion of the programming code for the processor abstraction is common as between at least some processor blocks and is shared by such processor blocks (see column 10, lines 29-40, which shows using C++ programming code and template classes to reuse or share common interfaces).

With respect to claim 9 (original), You further discloses the limitation wherein the programming code for the processor abstraction is organized into a tree form with generic code at a base node and more specific levels of code branching out at nodes therefrom, each processor block including at least one node from the tree (see FIG. 9, which shows the abstraction tree for address-related code; note that this tree applies to the processor type).

With respect to claim 10 (original), You further discloses the limitation wherein the engine further includes a high level portion for issuing generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions (see FIG. 2 and column 9, lines 28-45, which shows the high-level client interface used for issuing generic requests to a particular debugging server implemented for a particular debugging type and processor).

With respect to claim 11 (original), You further discloses the limitation wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type, wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor, and wherein the high level

portion issues generic request to the debugging type abstraction and to the processor abstraction to accomplish debugging actions (see column 10, lines 4-7, which shows that the portable debugging system uses abstraction; see also see FIG. 2 and column 9, lines 28-45, which shows the high-level client interface used for issuing generic requests to a particular debugging server implemented for a particular debugging type and processor).

With respect to claim 14 (original), You further discloses the limitation wherein the plurality of processor attributes supported by the processor blocks include processor attributes representative of members selected from a group consisting of an X86 processor family, an ALPHA processor family, and IA64 processor family, and combinations thereof (see FIG. 9, which shows support for X86 and 64-bit processor families, among others).

With respect to claim 15 (original), You further discloses the limitation wherein the debugger further has an executable for being executed by a user, for calling the engine, and for providing an interface between the user and the engine (see column 9, lines 28-45, which shows the client interface executed by a user for calling a particular debugging server or engine).

With respect to claim 18 (original), You further discloses the limitation wherein the particular debuggee is a dump file produced by a processor operating in a particular mode, wherein the debugging type attribute of the dump file corresponds to the particular mode, and wherein the particular debugging type block of the engine selected for debugging the dump file supports the debugging type attribute of the dump file (see column 6, lines 31-55, which shows the debugging types supported by the engine, including postmortem debugging for inspecting the

state of a program after it terminates, for example by using a dump file; note that the program state saved in the dump file would identify the processing mode, for example with an attribute).

With respect to claim 19 (original), You further discloses the limitation wherein the particular debuggee is a dump file produced by a type of processor, wherein the processor attribute of the dump file corresponds to the type of processor, and wherein the particular processor block of the engine selected for debugging the dump file supports the processor attribute of the dump file (see column 6, lines 31-55, which shows the debugging types supported by the engine, including postmortem debugging for inspecting the state of a program after it terminates, for example by using a dump file; note that the program state saved in the dump file would identify the type of processor, for example with an attribute; see also column 9, lines 17-26, which shows a plurality of processor attributes accounted for by the engine).

With respect to claim 20 (original), the claim recites the same limitations and features recited in claim 1. See the explanation for claim 1 set forth above.

With respect to claim 21 (original), the claim recites the same limitations and features recited in claim 2. See the explanation for claim 2 set forth above.

With respect to claim 22 (original), the claim recites the same limitations and features recited in claim 4. See the explanation for claim 4 set forth above.

With respect to claim 23 (original), the claim recites the same limitations and features recited in claim 5. See the explanation for claim 5 set forth above.

With respect to claim 24 (original), the claim recites the same limitations or features recited in claim 6. See the explanation for claim 6 set forth above.

With respect to claim 25 (original), the claim recites the same limitations and features recited in claim 8. See the explanation for claim 8 set forth above.

With respect to claim 26 (original), the claim recites the same limitations and features recited in claim 9. See the explanation for claim 9 set forth above.

With respect to claim 27 (original), the claim recites the same limitations and features recited in claim 10. See the explanation for claim 10 set forth above.

With respect to claim 28 (original), the claim recites the same limitations and features recited in claim 11. See the explanation for claim 11 set forth above.

With respect to claim 29 (original), the claim recites the same limitations and features recited in claim 15. See the explanation for claim 15 set forth above.

With respect to claim 32 (original), You discloses a method comprising:

(a) determining, for a particular debuggee, a debugging type attribute of the particular debuggee (see column 6, lines 31-55, which shows a plurality of debugging types supported by the debugging system; see also column 9, lines 28-33, which shows determining information from the host or debuggee such as, for example, the debugging type);

(b) selecting a particular debugging type block of an engine of a debugger for debugging the particular debuggee based on the determined debugging type attribute (see column 9, lines

28-33, which shows implementing or selecting an engine for a particular host environment, i.e. for a debuggee having a particular debugging type);

(c) determining, for the particular debuggee, a processor attribute of the particular debuggee (see column 9, lines 17-26, which shows a plurality of processor attributes accounted for by the engine; see also column 72, line 55 to column 73, line 5, which shows determining, for a particular debuggee, the architecture or the type of the processor);

(d) selecting a particular processor block of the engine of the debugger for debugging the particular debuggee based on the determined processor attribute (see column 9, lines 28-33, which shows implementing or selecting an engine for a particular host environment, i.e. for a debuggee having particular processor attributes); and

(e) employing the selected debugging type block and the selected processor block to debug the particular debuggee (see column 9, lines 28-45, which shows using a client interface for employing a debugging server or engine to debug a particular debuggee).

With respect to claim 34 (original), You further discloses the limitation wherein determining the processor attribute comprises sensing a particular type of processor from the debuggee (see column 72, line 55 to column 73, line 5, which shows determining or sensing the type of processor from the debuggee).

With respect to claim 35 (original), You further discloses the limitation wherein determining the debugging type attribute comprises sensing a particular type of debugging from the debuggee (see column 6, lines 31-55, which shows a plurality of debugging types supported

by the debugging system; see also column 9, lines 28-33, which shows determining or sensing information from the debuggee such as, for example, the debugging type).

With respect to claim 36 (original), You further discloses employing a high level portion of the engine of the debugger to issue generic requests to the selected debugging type block and to the selected processor block to accomplish debugging actions (see FIG. 2 and column 9, lines 28-45, which shows the high-level client interface used for issuing generic requests to a particular debugging server implemented for a particular debugging type and processor).

With respect to claim 37 (original), You further discloses the limitation wherein employing the high level portion comprises issuing generic requests from the high level portion to a debugging type abstraction and to a processor abstraction to accomplish debugging actions, the debugging type abstraction comprising a plurality of debugging type blocks and being available to provide debugging type services that vary in implementation for each debugging type, the processor abstraction comprising a plurality of processor blocks and being available to provide processor services that vary in implementation for each processor (see column 10, lines 4-7, which shows that the portable debugging system uses abstraction; see also see FIG. 2 and column 9, lines 28-45, which shows the high-level client interface used for issuing generic requests to a particular debugging server implemented for a particular debugging type and processor).

With respect to claim 38 (original), You further discloses running an executable of the debugger in response to a command from a user, the executable for calling the engine and for

providing an interface between the user and the engine (see column 9, lines 28-45, which shows the client interface executed by a user for calling a particular debugging server or engine).

With respect to claim 42 (original), the claim recites analogous limitations and steps to those recited in claim 32. See the explanation for claim 32 set forth above. Note that You also discloses a computer-readable medium having computer-executable instructions thereon, the instructions being organized into modules (see the title and abstract).

With respect to claim 44 (original), the claim recites analogous limitations and steps to those recited in claim 34. See the explanation for claim 34 set forth above.

With respect to claim 45 (original), the claim recites analogous limitations and steps to those recited in claim 35. See the explanation for claim 35 set forth above.

With respect to claim 46 (original), the claim recites analogous limitations and steps to those recited in claim 36. See the explanation for claim 36 set forth above.

With respect to claim 47 (original), the claim recites analogous limitations and steps to those recited in claim 37. See the explanation for claim 37 set forth above.

With respect to claim 48 (original), the claim recites analogous limitations and steps to those recited in claim 38. See the explanation for claim 38 set forth above.

7. Claims 16, 17, 30, 31, 33, 39, 40, 43, 49 and 50 are rejected under 35 U.S.C. 103(a) as being unpatentable over You, as applied to claims 15, 29, 32, 38, 42 and 48 above, respectively, and further in view of U.S. Pat. No. 5,533,192 to Hawley et al. (herein "Hawley").

With respect to claim 16 (original), although You discloses support for a plurality of debugging types (see column 6, lines 31-55), You does not expressly disclose the limitation wherein the executable includes an attribute that results in the selection of a particular debugging type block in the engine.

However, Hawley discloses an attribute in the executable used to select a particular debugger (see information 811 in FIG. 8a, which comprises the attribute; see also step 853 in FIG. 8b and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see the title and abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include, in the You system, an attribute as taught by Hawley, for the purpose of selecting the appropriate debugger, i.e. the appropriate debugging type block, in the absence of an alternative selection by the user (see Hawley, column 18, lines 26-37).

With respect to claim 17 (original), although You discloses support for a plurality of processor families or types (see column 9, lines 17-26), You does not expressly disclose the limitation wherein the executable includes an attribute that results in the selection of a particular processor block in the engine.

However, Hawley discloses an attribute in the executable used to select a particular debugger (see information 811 in FIG. 8a, which comprises the attribute; see also step 853 in FIG. 8b and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see the title and abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include, in the You system, an attribute as taught by Hawley, for the purpose of selecting the appropriate debugger, i.e. the appropriate processor block, in the absence of an alternative selection by the user (see Hawley, column 18, lines 26-37).

With respect to claim 30 (original), the claim recites the same limitations and features recited in claim 16. See the explanation for claim 16 set forth above.

With respect to claim 31 (original), the claim recites the same limitations and features recited in claim 17. See the explanation for claim 17 set forth above.

With respect to claim 33 (original), You does not expressly disclose the limitation wherein determining the debugging type attribute comprises receiving a selection of a particular type of debugging from a user.

However, Hawley discloses receiving a selection from a user for a particular debugger, i.e. for a particular type of debugging (see step 851 in FIG. 8b and column 18, lines 38-64), in a debugging system having a plurality of debuggers (see the title and abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to obtain, in the You system, a user selection as taught by Hawley, for the purpose of enabling the user to specify his or her preference for a particular debugger or type of debugging (see Hawley, column 18, lines 38-64).

With respect to claim 39 (original), although You discloses support for a plurality of debugging types (see column 6, lines 31-55), You does not expressly disclose the limitation

wherein the executable includes an identification of the debugging type attribute of the debuggee, and selecting the particular debugging type block in the engine based on the identification.

However, Hawley discloses an identification in the executable used to select a particular debugger (see information 811 in FIG. 8a, which comprises the identification; see also step 853 in FIG. 8b and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see the title and abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include, in the You system, an identification as taught by Hawley, for the purpose of selecting the appropriate debugger, i.e. the appropriate debugging type block, in the absence of an alternative selection by the user (see Hawley, column 18, lines 26-37).

With respect to claim 40 (original), although You discloses support for a plurality of processor families or types (see column 9, lines 17-26), You does not expressly disclose the limitation wherein the executable includes an identification of the processor attribute of the debuggee, and selecting the particular processor block in the engine based on the identification.

However, Hawley discloses an identification in the executable used to select a particular debugger (see information 811 in FIG. 8a, which comprises the identification; see also step 853 in FIG. 8b and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see the title and abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to include, in the You system, an identification as taught by Hawley, for the purpose

of selecting the appropriate debugger, i.e. the appropriate processor block, in the absence of an alternative selection by the user (see Hawley, column 18, lines 26-37).

With respect to claim 43 (original), the claim recites analogous limitations and steps to those recited in claim 33. See the explanation for claim 33 set forth above.

With respect to claim 49 (original), the claim recites analogous limitations and steps to those recited in claim 39. See the explanation for claim 39 set forth above.

With respect to claim 50 (original), the claim recites analogous limitations and steps to those recited in claim 40. See the explanation for claim 40 set forth above.

Conclusion

8. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Michael J. Yigdall whose telephone number is (703) 305-0352. The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2122

mjy

Hoanguyen Antony Nguyen Ba

ANTONY NGUYEN-BA
PRIMARY EXAMINER